NIST Special Publication 500-184

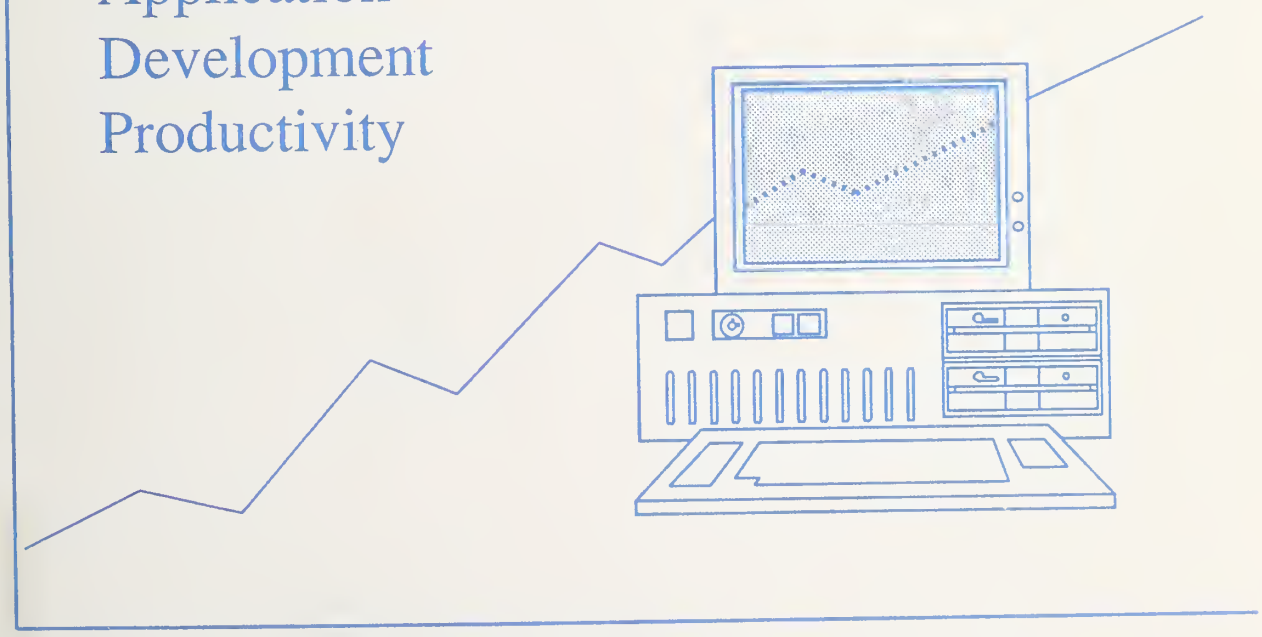
Computer Systems Technology

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

Functional Benchmarks for Fourth Generation Languages

Martha Mulford Gray
Gary E. Fisher

Application Development Productivity



NIST Special Publication 500-184

Functional Benchmarks for Fourth Generation Languages

Martha Mulford Gray
Gary E. Fisher

Computer Systems Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899

March 1991

U.S. DEPARTMENT OF COMMERCE
Robert A. Mosbacher, Secretary

NATIONAL INSTITUTE OF STANDARDS
AND TECHNOLOGY
John W. Lyons, Director

Reports on Computer Systems Technology

The National Institute of Standards and Technology (NIST) has a unique responsibility for computer systems technology within the Federal government. NIST's Computer Systems Laboratory (CSL) develops standards and guidelines, provides technical assistance, and conducts research for computers and related telecommunications systems to achieve more effective utilization of Federal information technology resources. CSL's responsibilities include development of technical, management, physical, and administrative standards and guidelines for the cost-effective security and privacy of sensitive unclassified information processed in Federal computers. CSL assists agencies in developing security plans and in improving computer security awareness training. This Special Publication 500 series reports CSL research and guidelines to Federal agencies as well as to organizations in industry, government, and academia.

National Institute of Standards and Technology Special Publication 500-184
Natl. Inst. Stand. Technol. Spec. Publ. 500-184, 60 pages (Mar. 1991)
CODEN: NSPUE2

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1991

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, DC 20402

PREFACE

This report has been prepared by the Computer Systems Laboratory (formerly the Institute for Computer Sciences and Technology). It is the fifth report issued in this area. The first report was a summary of the **Application Development Productivity Workshops**, held on November 13-15, 1985 at the National Institute of Standards and Technology (formerly the National Bureau of Standards.) Much of the discussion at this workshop was an urgent requirement for information and guidance on 4GLs. This impetus set the stage for accelerating research and the dissemination of information in this area.

The second report was **A Functional Model for Fourth Generation Languages**, NBS Special Publication 500-138, by Gary E. Fisher [FISH86], issued in June 1986. The purpose of this functional model was to define 4GLs "in a manner similar to specifying the functions of a specific software application." This definition allowed managers, technical personnel, and end-users to refer to commonly understood terminology in the 4GL context. In addition, it defined the interfaces between 4GLs, humans, operating systems, peripheral devices, and other application software.

The third report was **Guide to the Selection and Use of Fourth Generation Languages**, NBS Special Publication 500-143, by Martha Mulford Gray [GRAY86], issued in September 1986. The report provided guidance on the selection process in general for 4GLs; a description of the features, functions, and capabilities of 4GLs; and a brief discussion of the use of 4GLs. "Functional Benchmarks for Fourth Generation Languages" should be used with the guidance provided in [GRAY86].

The fourth report was **Application Software Prototyping and Fourth Generation Languages**, NBS Special Publication 500-148, by Gary E. Fisher [FISH87], issued May 1987. This report was designed as an introduction to the planning, organizing, executing, and controlling of a methodology for application prototyping. The report recommended that since 4GLs provide many of the capabilities necessary for prototype development, a combination of application prototyping and 4GLs can provide a cost-effective and controllable method of developing and maintaining software.

This fifth report represents years of research and the efforts of not only the authors but also of numerous staff and students who installed and tested ten 4GL products, executed and refined the benchmarking tasks, and assisted in the analyses of the results. We would especially like to acknowledge the efforts of Laura Benedict, Jon Berry, Dawn Hill, Nadine Melinger, Dan Nielsen, Teresa Stansbury, and Doug White.

TABLE OF CONTENTS

ABSTRACT

In recent years, fourth generation language (4GL) usage has expanded in data processing organizations, especially where end-users have assumed increased programming responsibilities. Since there are no 4GL standards, managers selecting a 4GL need some method of determining how well a particular 4GL will meet organizational, application, and user requirements. This report provides a methodology to assist in that determination. It contains "functional benchmarks" (as opposed to performance benchmarks) consisting of a testing methodology and descriptions of tests to evaluate the capabilities of a particular 4GL in relation to organizational requirements. Test results are evaluated at two levels: the 4GL's ability to perform a task, and the ease of performing it. These evaluations are combined with user-defined weighted requirements to produce an overall rating for each 4GL tested.

1. INTRODUCTION

The label "Fourth Generation Language" (4GL) has been applied to a wide variety of products. This variety makes any attempt to create generic evaluation measures difficult. Because each 4GL product is different, the most important measure is the product's ability to fulfill the requirements of the application at hand. This ability cannot always be determined from a vendor's demonstration or another user's opinion. The best method of evaluation is to test how well the 4GL satisfies the requirements of the application and to record the results.

NBS Special Publication 500-143, Guide to the Selection and Use of Fourth Generation Languages, provides guidance on the selection process for 4GLs; a description of the features, functions, and capabilities of 4GLs; and a brief discussion of the use of 4GLs. One of the steps (step nine) in the selection process described in that report is to "analyze the top few choices in detail using a benchmark, pilot test, demonstrations and/or user comments." This report describes functional benchmarks that could be used for this step. It contains a testing methodology and a set of functional benchmark tasks to assist managers in determining if a proposed 4GL meets an application's requirements.

1.1 Approach

The term "functional benchmark" was derived from the Federal Information Processing Standard, Guidelines for Benchmarking ADP Systems in the Competitive Procurement Environment, FIPS PUB 42-1.

"For these guidelines the term 'benchmarking' is used to convey the same meaning as the more explicit term 'benchmark mix demonstration.' A 'benchmark mix demonstration,' sometimes referred to as a Live Test Demonstration (LTD), consists of a user-witnessed running of a group (mix) of programs representative of the user's predicted workload on a vendor's proposed computer system in order to validate system performance. ***Another type of demonstration that is frequently called 'benchmarking,' more properly should be referred to as either a capability demonstration or a functional demonstration.*** *The latter type of demonstration is intended to show only system or functional capabilities in some specific areas without regard to total system performance."*

The methodology presented in this report is based on a study of fourth generation languages performed by the Computer Systems Laboratory (CSL), of the National Institute of Standards and Technology (NIST), formerly the National Bureau of Standards (NBS). To date, this research has generated a workshop summary and three publications: A Functional Model for Fourth Generation Languages, NBS Special Publication 500-138 [FISH86], Guide to the Selection and Use of Fourth Generation Languages, NBS Special Publication 500-143 [GRAY86], and Application Software Prototyping and Fourth Generation Languages, NBS Special Publication 500-148 [FISH87].

As part of this 4GL research, test databases were created and about 60 tasks developed and executed on various 4GL products to evaluate and measure characteristics of 4GLs. These specific tasks were modified and augmented to accommodate the diverse functionality provided by different 4GLs. A generalized benchmarking methodology for measuring the functionality (not performance) of 4GLs was developed from experience with these tasks and 4GL products.

The generalized tasks were rearranged and respecified to remove as much bias and implementation difficulty as possible. The improved set of tasks was then reordered in a manner that generally would require the least amount of work in performing tests of a specific 4GL. Most of the basic tasks were tested on microcomputer versions of the 4GL products but the generalized methodology and tasks should be applicable to any hardware that will support the 4GL products.

There are over 100 requirements and over 200 tasks included in this report. The time required to complete all of the tasks, if all of the requirements were needed, is difficult to estimate because it is dependent on the knowledge and expertise of the person running the tasks. If the person is familiar with the particular 4GL that is being tested, completing the tasks will require much less time than for someone with no 4GL experience because the testing time has to include the 4GL learning curve. We were able to complete all of the tasks in a few days for some of the products and a few weeks for others. Users with no 4GL experience might require twice as much time.

Since functional benchmarking is only part of the 4GL selection and evaluation process, the methodology described in this report will be most useful if used in conjunction with the selection process described in [GRAY86]. The selection process is summarized in the following section.

1.2 4GL Selection Process

Ten steps in the 4GL selection process are defined in [GRAY86] and illustrated in figure 1. The following presents a summarized version of the process and provides guidance on the selection process as a whole. The ten-step process should be completed whether you plan on purchasing a large, medium, or small system. The amount of depth and detail in each step will change depending on the size of the procurement.

The selection process begins with **step one**, a brief description of the problems that are to be solved or the applications that are to be implemented in the 4GL.

The **second step** is to complete an analysis of the application environments, i.e., the hardware, software, organizational, and user environments.

The **third step** is to decide how the selection decision will be made, i.e., if there will be a selection and evaluation team or a single decision-maker.

STEP	MAJOR CONCEPTS
1. DESCRIBE PROBLEM OR APPLICATION	o scope o purpose o domain
2. ANALYZE APPLICATION ENVIRONMENT	o hardware environment o software environment o organizational environment o user environment
3. DECIDE ON SELECTION APPROACH	o establish selection and evaluation team o identify decision makers and control points
4. DEFINE REQUIREMENTS	o identify application requirements
5. DEVELOP LIST OF DESIRED 4GL FEATURES	o use Product Selection Criteria
6. RATE DESIRED FEATURES	o establish users rating system o identify mandatory features o identify undesirable features
7. SELECT CANDIDATE PACKAGES	o survey literature o survey software reference services o attend conferences, trade shows
8. RATE 4GLs AND SELECT TOP FEW	o screen using hardware, operating systems, and mandatory features o use other ratings to narrow selection
9. ANALYZE TOP FEW IN DETAIL	o benchmark o pilot test o demonstrations o user comments
10. SELECT 4GL	

Figure 1. 4GL Selection Process

The **fourth step** is to define the requirements for the 4GL based on the information gathered in the first two steps. Emphasis must be placed here on defining actual requirements, and not on unrealistic demands, wishes, etc.

After the requirements have been defined, a list of 4GL features which can fulfill these requirements must be developed. (See [GRAY86] for a full description of these features.) This is **step five** of the selection process.

Step six is to develop a ranking or rating system for the 4GL features in order to differentiate between the most desired and the least desired features.

Step seven consists of selecting candidate packages. Information on the availability of 4GLs can be obtained from computer literature, software reference services, trade shows, computer conferences, and computer user groups.

Step eight is to eliminate most of the candidate packages. First, a screening process should be established to eliminate packages that do not fulfill the hardware, operating system, or mandatory feature requirements, or that possess undesirable qualities. After the screening process, further elimination of products should be based on rating the other features of the 4GLs and comparing these ratings with the user ratings of the desired features. This step should eliminate all but a few products.

Step nine is to analyze the top few candidate products in detail. Approaches such as benchmarking, pilot testing, vendor demonstrations, and gathering user opinions can be used for this step.

The final step, **step ten**, is to select the most appropriate package.

2. FUNCTIONAL BENCHMARKING METHODOLOGY

The functional benchmarking methodology is designed to determine if a 4GL has the functionality necessary to meet the requirements of an application. The requirements are dependent on what needs to be done and by whom; specifically, the characteristics of the application, and the characteristics of the users who will be working with these applications. These requirements must be determined before the functional benchmark tasks can be performed.

The second step of the selection process in [GRAY86] is to analyze the application environment. A process is provided in [GRAY86] for analyzing this application environment, which includes the hardware, software, organizational, and user environments. The results of this analysis are used in step four to define the application requirements. For example, a typical requirement in data processing environments is the need for the 4GL to provide programming interfaces for both casual and expert users. For the casual user, a menu-oriented or a fill-in-the-blanks type of interface may be required. For the expert or professional user, a succinct command language may be needed. Not all 4GLs provide both types of interfaces. Those that do may not necessarily provide all of the functionality needed for the application.

These application requirements are used to develop a list of desired 4GL features, step five of the selection process. These features must be rated by importance for the application and weights must be developed for this rating. Step six includes identifying mandatory or critical features that the 4GL must possess. These mandatory or critical features can be used to "narrow the field," eliminating some of the potential 4GL products. The functional benchmark tasks are run on the few products that remain after eliminating those that do not possess the mandatory or critical features. The results of these tasks are compared with the weights of the selection features to provide the final selection guidance.

The functional benchmarks are divided into numerous tasks. Each task should be evaluated on the 4GL's ability to perform the task and the ease of performing the task. Determining if a 4GL can perform the task is fairly objective; however, judging its ease of use is more subjective. While objectivity in developing these tests was a significant goal, we found that subjective measures were also needed. Judgements such as a preference for function key-initiated actions as opposed to

spelled-out command-initiated actions, or color monitor screens rather than black and white screens, are normally realized after experience has been gained and are essentially subjective in nature.

Critical requirements and the required level of functionality provide the objective measures of these tasks. If a requirement is critical (i.e., the 4GL must provide the functionality), then a 4GL that does not provide it objectively fails the test for that capability. If the capability is available, but cannot be used effectively (e.g., procedures may contain conditional statement constructs such as IF statements, but contain no IF-THEN-ELSE constructs), then you may decide that the 4GL does not meet the threshold functionality required, and it may fail the test.

The functional benchmark tasks were designed to run on a microcomputer system with sufficient equipment to allow the 4GL to be installed. They could, however, be run on a larger computer if desired. We have endeavored to make the tasks as generic as possible to remove portability problems due to systems software, architecture and other dependencies.

While a production system based on the selected 4GL may eventually execute on a minicomputer or mainframe, the microcomputer versions generally have sufficient capabilities to allow testing of functionality and provide a cost effective instrument for doing so. Once a 4GL has been selected for an application, it should be tested on the hardware that will be used before it is put into production.

2.1 Benchmarking Process

There are six steps in the process of performing these benchmark tasks. They are--

1. Rate the desired features of a 4GL according to the application's requirements and assign appropriate weights to these features.

2. Select the functional benchmark tasks which are appropriate for measuring these desired features.

3. Perform the appropriate tasks.

4. Evaluate the 4GL's performance on each task.

5. Compute the 4GL's weighted score.

6. Analyze the results.

2.1.1 Rate 4GL Features and Assign Weights

The importance of a particular requirement is determined by its weight in relation to other requirements. You assign these weights as your preference before tests are performed. For example, if a relational database capability is critical for your applications, the weight associated with relational capabilities will rank very high on your preference list. On the other hand, graphics capabilities may play no role in your requirements. You would, therefore, assign little or no importance to this particular aspect.

The assignment of weights to requirements may be performed in any manner deemed appropriate to an organization's philosophy and culture. Some organizations provide detailed lists of requirements to those who will use the system and ask for ranked preferences for each requirement. Others may do this ranking by committee. As the number of requirements increases, ranking these requirements gets much more difficult.

Another suggested method of assigning weights consists of using a generalized requirements checklist like the one provided in this report (see sec. 3). Weights of importance are assigned using any numbering system. Often numbers from one to five are used with the number five having near-critical importance and the number one having little importance. One user or several users could assign the weights. If several users assign weights, an individual must be assigned to develop one overall or average weight for each requirement. Requirements that are not germane would be rated zero.

2.1.2 Select Appropriate Tasks

You should identify which tasks are to be performed depending on the requirements that you have specified. The tasks that relate to the requirements are identified by section number attached to each requirement. The number of tasks that should be run are totally dependent on the identified requirements of the application. If the application does not require specific functions, these tasks should not be executed.

2.1.3 Perform the Appropriate Tasks

We suggest that you run the tasks in the order they are presented. For example, the 4GL must be installed before files or tables can be defined using the 4GL. We strongly suggest that you adjust the tasks to meet the specific requirements of your application and use files required by the application.

2.1.4 Rate the Performance of Each Task

As you perform each task, you should rate the results according to how much of the task was accomplished and how well the 4GL performed the task. There are two numeric ratings given to each result. The first rating is the functional rating, or how much of the task was accomplished (e.g., 0-none, 1-some, 2-most, 3-all). The second describes how well you think the 4GL performed the task or how easy it was for you to perform the task (e.g., 0-not done/impossible, 1-poorly/hindrance, 2-adequately/helpful, 3-easily/very helpful).

Some of the tasks may seem difficult to rate in relation to functionality and ease of use/performance. Documentation, for example, may not appear to have a measurable performance, but if it is disorganized or incomplete, you may find that it hinders progress or may offer no help at all. In this case, it would have a direct impact on performing a specific test.

Since the tasks are grouped by requirement, some requirements have more than one task. In some cases this is because the requirement cannot be tested using one task but takes multiple steps or multiple tasks. An example of these multistep tasks would be importing ASCII files. First a database file structure must be described, data imported, and the imported data compared with the

original data. Each of these tasks can be rated but they are grouped under the requirement of "Ability to import ASCII files." Others require testing various aspects of the requirement to obtain a thorough testing of several variations. For example, the requirement "Reporting options for user specified page breaks" has four tasks: generate a report with a page break after each total, after each subtotal, after 40 lines, and after a sort value has changed. There is not a single task that adequately tests page breaks.

After you have rated these multiple tasks under a requirement, you need to average the ratings to get one average value for functionality and one for ease of use for this requirement. Add the functionality ratings for each task and divide this number by the number of tasks performed. Repeat for the ease of use ratings. Enter these numbers on the average line.

2.1.5 Compute the Weighted Score

Once the selected tasks for a requirement have been performed, the scores are totaled, averaged, and multiplied by the weight given for the requirement. These weighted numbers are then added together to produce a total weighted value for the 4GL.

2.1.6 Analyze the Results

These procedures are repeated, with the same weighted requirements, for all 4GL products under consideration. The total weighted value for each 4GL tested will provide an indication of which 4GL will most closely match your stated requirements. The highest scoring product should fulfill the majority of your requirements.

In addition to looking at the total numbers you should also look at the ratings of each functional area and of the individual requirements, especially for the highest weighted requirements. This is important when two products have total scores that are very close to each other. If a 4GL is rated somewhat higher than another, but you do not feel as comfortable about the higher rated one, you may be able to determine the reasons for this by analyzing the scores for individual functional areas of each 4GL. An analysis of variations of individual area scores may tell you how much confidence you should have in your choice.

2.2 Benchmarking Organization

The requirements and tasks are divided into several functional areas that closely parallel the feature areas described in [GRAY86]:

1. operational features;
2. data management features;
3. user interfaces;
4. report writing features;
5. language features;
6. graphics features;
7. security features;
8. implementation features; and
9. application development.

This organization was designed to facilitate the users' normal usage of a 4GL. The software cannot

be run until it is installed, data cannot be queried until the database exists, etc. This logical grouping of functional areas can be described as follows.

2.2.1 Operational Features

The operational features are those functions that are basic to the operation of the 4GL. In general, operational features include hardware and software requirements for a 4GL, portability attributes, performance characteristics, communication features and operating system interfaces. Since some of these features are critical or mandatory features, products that do not meet these requirements should be eliminated before getting to the benchmarking step of the selection process (e.g., products that will not run on the hardware required would not get to the benchmarking step). Thus, tasks for these features are not included in this report.

There are other operational features that cannot be evaluated before a database is created and experience gained with the 4GL. These include such features as diagnostic messages, interfacing the 4GL to other programming languages, and quality of the documentation. These features are included in the tasks for implementation features.

The operational features included in this section of requirements and tasks are those that relate to getting a 4GL up and running: installation and documentation of the installation procedures.

2.2.2 Data Management Features

The general data management functions include the capabilities necessary to manage and manipulate data; to define data structures, store data, and retrieve data. Because a 4GL has other capabilities to manipulate data such as query languages, user interfaces, report writing features, etc., this section is limited to data definition features, file creation, importing, and exporting data.

The ability to easily define file structures, data formats, and required data types is important to the capability of creating new databases. The ability to load data from external files into the database can be critical for converting from another application or in accepting data created by other applications. Since many 4GLs can process data from many database systems without conversion, the ability to read, write, and query from external files is also important to many applications. These functions are included in this section.

2.2.3 User Interfaces

There are many different kinds of user interfaces to 4GLs. Menus and screens are some of the primary means that a user has to interact with the 4GL and an application. Some language features are also part of the user interface but they are included in another section.

The ability to design screens and menus is required for most application development. The 4GL may need to provide functions such as field character validation, field masks, required fields, filled fields, computed fields, default field values, screen/data field refresh, interfield checks, and interrecord checks for the application. The requirements and tasks included under user interfaces cover the fundamental aspects of screen and menu design.

2.2.4 Report Writing Features

The ability to display or print both detailed and summarized data in a straightforward and meaningful manner is one of the most important features of a 4GL. Generally, a 4GL provides a default report format for simple reports so that the user can issue a query and have the results printed as a report without having to specify elements of the report such as paging, date, page break, column position, etc. In addition, the 4GL provides capabilities for specifying precisely these elements, if the user desires. The most complete report writer will provide the most flexibility for displaying data.

2.2.5 Language Features

There are many language functions of a 4GL: nonprocedural language functions, query language functions, programming or command language functions, interfaces with other programming languages, and the ability to produce compiled or optimized code. All of these are very important to the operation and functionality of the 4GL. Quite often, the language capabilities determine whether the 4GL has the power or ease of use that is desired for the application. If the user interface is difficult to use, the 4GL may not be appropriate for novice users. If it lacks some of the programming capabilities of a third generation language, it may not be powerful enough for the application. The richness of the language and the level of integration between all aspects of the 4GL (report writer, screen generator, query language) strongly influence whether or not the software is in fact a fourth generation language.

The language in which the 4GL is written (the implementation code) and the type of application code it produces may influence the portability of the application and the performance of the 4GL. These features of the language are difficult to generalize into tasks but should be considered in the overall assessment of the 4GL.

In many cases, the nonprocedural language of the 4GL is implemented in screens and menus. Because these are also used for data input, display, etc., the tasks for testing these functions are included in the user interface. This section includes requirements and tasks for query language functions, nonprocedural language functions, procedural language functions, and integration aspects of the language.

2.2.6 Graphics Features

Some 4GLs have graphics facilities integrated with the other functions of the 4GL, while others provide interfaces to separate graphics software. Some 4GLs simply provide a third generation language interface that allows existing graphics software, such as FORTRAN routines, to be executed. The requirements of the application have to be matched with these capabilities.

There is a wide range of graphics capabilities that are available. Some 4GLs only provide simple business graphics capabilities, bar charts, pie charts, and line graphs. Others may provide three-dimensional graphs, scatter diagrams, and logarithmic graphs. The most important feature of any of these capabilities may be the amount of integration with the rest of the 4GL functions. For example, creating a graph whose data points are obtained directly from the database and whose labels are obtained from the data dictionary, requires much less effort than one which requires

typing all labels or re-entering the data.

Novice users may need graphics features to perform scaling, calculating percentages for pie charts, placing labels, and formatting bar placement. Users may want the ability to view the chart on the screen and manipulate the results before any printing takes place. Some users may want high-quality presentation graphics, high resolution capabilities, color, multiple fonts, and interfaces to plotters or photographic output devices.

The requirements and tasks given for graphic features are not meant to cover all possible graphic requirements. If an application has very complex graphic requirements, additional tasks should be added for the benchmarking process.

2.2.7 Security Features

The security features of a 4GL often include user identification and password protection at various levels (user, owner, file or table, record, field, entity or data item, procedure [edit, delete, insert], etc.), backup protection and protection of data during crashes, encryption capabilities, and features for audit trails. The level of security required has to be determined for the application and the system the application runs on. Often the system security or pre-existing software provides some of the needed security features. If adequate security software already exists, then the software attributes of the 4GL have to be tested for secure interfaces to ensure that they do not provide a "back door" into the system. If the security of the system is dependent on the security attributes of the 4GL then these features of the 4GLs should probably be more thoroughly tested than the tasks that we provide. The tasks given for this area cover "typical" kinds of security functions for 4GLs that would run in a microcomputer environment.

2.2.8 Implementation Features

The tasks for these features are tasks that cannot be evaluated until the user has gained experience with the 4GL. As mentioned in section 2.2.1, many of these tasks are related to operational features such as interfacing with other languages, interfacing with the operating system, quality and usefulness of diagnostic messages, and on-line help capabilities. Others are related to overall usage of the 4GL, such as judging the quality of the documentation.

2.2.9 Application Development

The tasks for these features are really tasks which combine or incorporate different functions of the 4GL into a cohesive application. The application may be generated from the screen painter, report generator, and language features which have been tested in other feature areas. This grouping of features, however, is meant to test the ability of the 4GL to put them all together to build an application. Language integration, language functionality, debugging aids, and programming messages are examples of 4GL features that really cannot be tested except by building an application or a prototype of an application.

3. 4GL REQUIREMENTS CHECKLIST

The 4GL requirements checklist must be used for the first procedure in the benchmarking process and can also be used for step six of the 4GL selection process, "develop a ranking or rating system for the 4GL features so that the most desired features can be differentiated from the least desired" [GRAY86]. If the checklist is being used for step six, mandatory features should be identified in this step plus any features which would make a package unacceptable. First, the features of a 4GL that are critical for an application must be identified. We suggest that these features be identified with an asterisk (*), meaning that the application would not be acceptable without these features. This also implies that a product would be unacceptable for use if it did not possess these features. The asterisks represent the highest rating or weighting that a feature can receive and should be used to "weed out" those products that do not meet the application's critical requirements.

If the checklist is being used for the benchmarking process, the critical features have already been used to eliminate those products that do not meet the mandatory requirements. At this point an asterisk is probably not the most useful indicator. The critical features should simply be assigned the highest rating.

The importance of the other 4GL features relative to the application also has to be determined. Weights for these other features should be assigned. We suggest a scale of one to five with five representing the highest rating. Features with a weight of five are most important to the application. Features with a weight of one are not very important. These might be nice to have but are not the most important features that are required of the application. A weight of zero would imply that this feature is not required by the application.

ASSIGNMENT OF WEIGHTS FOR APPLICATION

	Weight
1. Operational Features	
A. Installation management	_____
B. Ease of installing on multiple platforms	_____
2. Data Management	
A. Basic data dictionary	_____
B. Synonym or aliases	_____
C. Headings, titles, or labels	_____
D. Comments or remarks	_____
E. Date formats	_____
F. Time formats	_____
G. Money formats	_____
H. Variable-length fields	_____
I. Capable of handling _____ # chars per field	_____
J. Capable of handling _____# fields per record	_____
K. Keyed or indexed files	_____
L. Data structure management	_____
M. Ability to read, query, or report from ASCII files	_____
N. Ability to import ASCII files	_____
O. Ability to export ASCII files	_____

P. Ability to read, query, or report from _____ formatted files _____
Q. Ability to import _____ formatted files _____
R. Ability to export _____ formatted files _____
S. Ability to modify files while loading _____
T. File check while loading _____
U. File modification _____

3. User Interface
A. Capability to create simple menus _____
B. Capability to create complex menus _____
C. Default data entry screens provided _____
D. Capability to create straight text screens _____
E. Capability to create display screens _____
F. Data checking on data entry _____
G. Reverse video control _____
H. Blinking data entry fields _____
I. Calculated fields display _____
J. Display of HELP information on screen _____
K. Capability to generate more complex screens _____
L. Ability to print screen layout document _____

4. Report Writing Features
A. Default report formats provided for paging, dating, and headings _____
B. Default report formats provided for summary functions _____
C. Default report selections provided for ascending sorts _____
D. Default report selections provided for descending sorts _____
Complex reporting options for:
E. Row totals and subtotals _____
F. Percent formats _____
G. Zero suppression _____
H. Floating dollar signs _____
I. Comma option for numeric fields _____
J. User specified spacing _____
K. User specified page breaks _____
L. Table lookups _____
M. Headings _____
N. Footers _____
O. Font selection _____
P. Odd size printouts _____
Q. Forms printing _____
R. Suppressed fields _____
S. Calculated fields _____
T. Specific field placement _____
U. Output device independence (i.e., user could select screen, printer, or file) _____
V. Output device adjustments (i.e., user selects screen, printer, or file and 4GL adjusts for page size, screen size, etc.) _____

5. Language Features

A. SQL implementation _____
B. Simple queries on one file _____
C. AND OR logic - one file _____
D. Relational join _____
E. AND OR logic - two files _____
F. Complex Boolean logic _____
G. Traps command or statement errors _____
H. Facilitates correction of errors _____
I. Compiles code _____
J. Interprets code _____
K. Condition logic _____
L. Looping logic _____
M. Intrinsic date/time functions (DATE, DAY, MONTH, etc.) _____
N. Intrinsic numeric functions (MINIMUM, MAXIMUM, AVERAGE, etc.) _____
O. Statistical functions:
    Standard deviation _____
    Variance _____
    Regressional analysis _____
    Correlation _____
    Discriminant analysis _____
    Factor analysis _____
    Descriptive statistics _____
P. Ability to process missing data _____

6. Graphics Features

A. Ability to create bar graphs _____
B. Ability to create pie charts _____
C. Ability to create exploded pie charts _____
D. Ability to create scatter diagrams _____
E. Ability to create line graphs _____
F. Ability to create connected point plots _____
G. Ability to incorporate trend lines in scatter diagrams _____
H. Ability to curve fit or smooth _____
I. Availability of automatic scaling _____
J. Availability of logarithmic scaling _____
K. Ability to generate field labels and titles from the data dictionary _____
L. Ability to utilize multiple fonts _____
M. Ability to calculate values to incorporate in the graph _____
N. Grid option _____
O. Color option _____
P. Hatch pattern options _____
Q. Ability to process missing values _____

7. Security Features

A. Ability to specify user access _____

B. Ability to specify file security _____
C. Ability to limit command or procedure usage _____
D. Ability to specify record access _____
E. Ability to specify field access _____
F. Ability to encrypt/decrypt files _____
G. Ability to encrypt/decrypt data dictionary _____
H. Default logon procedure _____
I. User profile _____
J. Ability to define system administrator _____
K. Feature restrictions _____
L. Ability to create a user logon audit file _____

8. Implementation Features
A. Interface to other software packages _____
B. Interface to other languages _____
C. Interface to operating system _____
D. POSIX compatibility _____
E. Online HELP _____
F. Documentation _____
G. Interface to external editor/word processor _____
H. Error messages and warnings _____

9. Application Development
A. Program editor _____
B. Error messages _____
C. Branching logic _____
D. Looping logic _____
E. Language integration _____
F. Ability to call subroutines _____
G. Ability to imbed queries in procedures _____
H. Ability to share temporary data _____
I. Ability to create global variables _____
J. Ability to create local variables _____
K. Ability to define calculations using field values _____

4. BENCHMARKING TASKS

The functional benchmarks are divided into numerous tasks. The tasks are organized by the groupings described previously. The tasks are listed by the application requirement number, following the same order as the requirements checklist.

Each task should be evaluated on the 4GL's functional ability to perform the task and the ease of performing the task. We suggest a rating scale of zero to three for this process. For functionality, a 4GL that could perform all of the functionality that was required to accomplish the task would receive a three. If most of the task could be accomplished, a two; some, a one; and if none of the task could be completed, the 4GL would score a zero. For ease of use/performance, if the task was completed easily and the 4GL facilitated the process, the 4GL would score a three. If the task was extremely difficult to accomplish the 4GL might score a one. If it was impossible to complete,

it would score a zero. A summary of this suggested rating scale is provided below and at the beginning of each section.

	Functionality	Performance/Ease of use
RATING	**0-none**	**0-not done/impossible**
SCALE	**1-some**	**1-poorly/hindrance**
	2-most	**2-adequately/helpful**
	3-all	**3-easily/very helpful**

4.1 Operational Features

Any product must be installed before you can test its capabilities. Some products are much more difficult to install than others. Because all products are different, we cannot specify commands for a specific type of installation. You must follow the directions supplied with the product.

Your goal during this process is to evaluate the difficulty of the installation, how well the documentation matches what actually occurs, if the documentation provides enough detail, if the storage requirements are as advertised, and how difficult it is to recover from any errors that may occur during installation.

Some applications may require individual installation at numerous sites with different personnel performing installation. In this case, even more attention should be paid to evaluation results than what we suggest. For applications that will require only one installation and occasional upgrades, the additional tasks for multiple installation sites should not be required.

	Functionality	Performance/Ease of use
RATING	**0-none**	**0-not done/impossible**
SCALE	**1-some**	**1-poorly/hindrance**
	2-most	**2-adequately/helpful**
	3-all	**3-easily/very helpful**

1A. Installation management

	FUNC.	EASE
1. Install the 4GL.	____	____
2. Rate the documentation (user's guide, reference manual, etc.) for installation instructions.	____	____
3. Rate the ability to customize the installation. (i.e., does the user have the option on whether or not to include certain features, such as the demonstration/tutorial or the graphics feature?)	____	____
Total	____	____
Average (Total/3)	____	____ (1A)

1B. Ease of installing on multiple platforms

Install the 4GL on a second system with a different operating system or different hardware. ____ ____ (1B)

4.2 Data Management

A 4GL contains many capabilities for managing and manipulating data. The first of these is the definition of data structures to be stored in the database. This set of tasks will help determine if the 4GL contains appropriate functionality to define and manage the data structures needed for the planned applications.

The second major capability under the data management heading is that of data loading. A separate section is devoted to testing this aspect. The ability to load data from external files into the database will assist in converting from another application or in accepting data created by other applications. We suggest that data from an existing application be used to perform the data loading tests. Since the personnel performing the evaluation should be familiar with that data, errors in test performance will be more readily visible than if "made-up" data were used.

If no specific data comes to mind for performing these tests, we suggest generating at least two files containing data appropriate for the planned applications. We suggest using a large file with two to three thousand records, and a smaller file, with each record containing at least several alphanumeric data elements and several types of numeric data elements.

At first glance, it may appear that some of the tests can be combined. However, evaluators may run into problems with different 4GLs due to limitations of each particular 4GL in performing combined tasks. Errors encountered in evaluation may be due to combinations of effects. In such cases, it may be difficult, if not impossible, to ascertain what the real problem is and how to overcome it. For this reason, the tests are specified individually to control as much external influence as possible.

	Functionality	Performance/Ease of use
RATING	**0-none**	**0-not done/impossible**
SCALE	**1-some**	**1-poorly/hindrance**
	2-most	**2-adequately/helpful**
	3-all	**3-easily/very helpful**

2A. Basic data dictionary **FUNC. EASE**

1. Using whatever method is suggested by the vendor (e.g., screens, forms, prompts, etc.), define a sequential file (FILE A). ____ ____
2. Define a file with at least one key or index made up of at least one data element (FILE B). ____ ____
3. Define a numeric data element containing a decimal point. ____ ____
4. Define an integer data element. ____ ____
5. Define a data element containing only alphabetic characters. ____ ____
6. Define a data element containing both alphabetic and numeric characters. ____ ____

7. List all information pertaining to FILE A and FILE B including the file name, file organization, data element names, data element types, and sizes on the terminal screen. ____ ____

8. List all information pertaining to FILE A and FILE B including the file name, file organization, data element names, data element types, and sizes on a printer. ____ ____

Total ____ ____

Average (Total/8) ____ ____ (2A)

2B. Synonym or aliases
Define a synonym or alias for a data element in FILE A. ____ ____ (2B)

2C. Headings, titles, or labels
Define a heading, title, or label for a data element in FILE A. ____ ____ (2C)

2D. Comments or remarks
Define a comments or remarks section for the file or a data element in FILE A. ____ ____ (2D)

2E. Date formats
Define a data element in FILE A with a date format. ____ ____ (2E)

2F. Time formats
Define a data element in FILE A with a time format. ____ ____ (2F)

2G. Money formats
Define a data element in FILE A with a money format. ____ ____ (2G)

2H. Variable-length fields
Define a data element in FILE A with a variable length. ____ ____ (2H)

2I. Capable of handling _____# chars per field
Define a numeric element with the size of the largest number that will most likely be used in an application or a character element with the largest number of characters that will be used. ____ ____ (2I)

2J. Capable of handling _____# fields per record
Define a sufficient number of data elements in FILE A and FILE B to determine if the 4GL can accommodate the largest number of fields required per record. ____ ____ (2J)

2K. Keyed or indexed files

1. Define a sequential file (FILE C). ____ ____

2. Define a file with at least one key or index made up of at least one data element (FILE D). ____ ____

3. Define a key in FILE D that is the same as the key in FILE C. ____ ____

4. Define a key in FILE D that is different from all previous keys. ____ ____

5. Define a key in FILE C that is made up of at least two concatenated data elements. ____ ____

Total ____ ____

Average (Total/5) ____ ____ (2K)

2L. Data structure management

1. Delete a non-key data element from FILE D. ____ ____

2. Delete a key data element from FILE D. ____ ____

3. Delete a data element from FILE C. ____ ____

4. Delete FILE C. ____ ____

5. Copy FILE A description to FILE E. ____ ____

6. Copy FILE B description to FILE F. ____ ____

7. Delete FILE E and FILE F. ____ ____

8. Edit a data description by changing the spelling of a field/data element name. ____ ____

9. Change the format of a field/data element. ____ ____

Total ____ ____

Average (Total/9) ____ ____ (2L)

2M. Ability to read, report, or query from ASCII files

1. Define a file structure for an external ASCII file (FILE 1) to be loaded. ____ ____

2. Read the external file and create a report or query from the external file's data without creating a 4GL formatted file. (If you could not create a report or query from the external file, list the first 10 and last 10 records of the external file using any means appropriate.) ____ ____

Total ____ ____

Average (Total/2) ____ ____ (2M)

2N. Ability to import ASCII files

1. Define a file structure for a 4GL database that will accommodate FILE 1. ____ ____

2. Load the data from the external file into the 4GL database structure creating the 4GL internal file, FILE 2. ____ ____

3. To ensure that the data is loaded correctly, repeat the same report or query that you created in task 2M and compare the data. (If you could not create a report or query from an external file, list the first 10 and last 10 records of the database file (FILE 2) using the 4GL. Compare the data in this report with the data in the external file.) ____ ____

Total ____ ____

Average (Total/3) ____ ____ (2N)

2O. Ability to export ASCII files
Save the results of the previous query to an external ASCII file. Type or print the external file to compare results. ____ ____ (2O)

2P. Ability to read, report, or query from _____ formatted files
Repeat tasks 2M for all file formats required.
____ ____ (2P)

2Q. Ability to import _____ formatted files
Repeat tasks 2N for all file formats required. ____ ____ (2Q)

2R. Ability to export _____ formatted files
Repeat tasks 2O for all file formats required. ____ ____ (2R)

2S. Ability to modify files while loading

1. Define an internal file (FILE 3) structure similar to the external file structure of FILE 1, and augment it with at least one data element that is not in the external file structure. ____ ____

2. Load data from the external file (FILE 1) into the augmented 4GL database file (FILE 3). ____ ____

3. Define an internal file (FILE 4) structure similar to the external file structure of FILE 1, but rearrange the order of the data elements. ____ ____

4. Load the data from the external file (FILE 1) into the rearranged 4GL file (FILE 4). ____ ____

Total ____ ____

Average (Total/4) ____ ____ (2S)

2T. File check while loading

Load descending sorted data from the external file into an ascending sorted internal file without first deleting the data from the internal file. (Rate the functionality 3, if the duplicates are rejected.) ____ ____ (2T)

2U. File modification

1. Display basic attributes of a file, including the file size, number of records, time the file was created, time the file was last modified or accessed. ____ ____
2. Display file usage information including number of records added, deleted, or modified. ____ ____
3. Add a record to the file. Display the record. ____ ____
4. Modify a record in the file. Display the record. ____ ____
5. Delete a different record in the file. ____ ____
6. List the attributes of the file including the file size, number of records, time the file was created, time the file was last modified or accessed. Make certain that the changes were reflected in the listing. ____ ____
7. Display the file usage information including number of records added, deleted, or modified to make certain that the changes were reflected in the listing. ____ ____

Total ____ ____

Average (Total/7) ____ ____ (2U)

4.3 User Interface

Screens and menus are the primary means that a user has of interacting with an application. For this reason, a 4GL must have a diversified and robust set of screen formatting capabilities. These capabilities are generally grouped in two categories: screen formatting commands and data manipulation commands. Screen painters are often used to implement these capabilities within a 4GL. Text, symbols, and data input/output areas are defined on a terminal screen through placement of a cursor and the use of predefined commands or control keys.

	Functionality	Performance/Ease of use
RATING SCALE	0-none	0-not done/impossible
	1-some	1-poorly/hindrance
	2-most	2-adequately/helpful
	3-all	3-easily/very helpful

3A. Capability to create simple menus **FUNC. EASE**

1. Define a default menu with at least two selections. ____ ____

2. Define a second default menu to be executed as one of the choices of the first menu. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (3A)

3B. Capability to create complex menus

1. Define a vertical menu containing at least two selections. ____ ____
2. Define a horizontal menu containing at least two selections. ____ ____
3. Define a menu that uses function keys or control keys as the means of selecting. ____ ____
4. Define a menu that uses cursor positioning as the means of selecting. ____ ____
5. Define a menu that uses character strings or numbers as the means of selecting. ____ ____
6. Define a pull-down menu with at least two selections. ____ ____
7. Define a menu with text, menu background, and menu border in different colors. ____ ____
8. Define a menu with more entries than can be displayed on the screen at one time. ____ ____
9. Define a menu that scrolls according to the location of the selection cursor (i.e., on a vertical menu, some of the selections may not be visible through scrolling of the selections off the top or bottom of the menu. A horizontal menu may scroll to the left and right). ____ ____
10. Define a menu that allows speed searching of the selections (i.e., the selection cursor advances to the next selection starting with the character entered by the user). ____ ____

Total ____ ____

Average (Total/10) ____ ____ (3B)

3C. Default data entry screens provided

Access a default data display screen containing at least two data elements and the names associated with the data elements. ____ ____ (3C)

3D. Capability to create straight text screens

Define an information screen containing text only (no records or variables). ____ ____ (3D)

3E. Capability to create display screens

1. Define a screen and associated process to display a record in a file when the screen is executed. ____ ____ (3E)

3F. Data checking on data entry

1. Access a default data entry screen containing at least two data element input/display areas and the names associated with the data elements. (These data elements may be data elements defined in a previous test for file definition or data loading.) ____ ____

2. Enter an alphanumeric string in a numeric data entry area to ascertain if the data is accepted or not. ____ ____

3. Enter a numeric string in an alphanumeric data entry area to ascertain if the data is accepted or not. ____ ____

4. Enter a non-date string in a date entry area to ascertain if the data is accepted or not. ____ ____

5. Using a data element for which an entry is required, try to bypass the field without entering a value. ____ ____

6. Using a data element for which a default entry is defined, bypass the entry field without entering a value to determine if the default value has been automatically displayed. ____ ____

Total ____ ____

Average (Total/6) ____ ____ (3F)

3G. Reverse video control

Define a reverse video data entry area. ____ ____ (3G)

3H. Blinking data entry fields

Define a blinking data entry area. ____ ____ (3H)

3I. Calculated fields display

Define a data display area with the contents computed from other data entry areas (i.e., such as a total, balance, or extended amount). ____ ____ (3I)

3J. Display of HELP information on screen

1. Define a data display window that contains help information for a particular data entry area. ____ ____

2. Define a help display area on screen. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (3J)

3K. Capability to generate more complex screens

1. Define a screen containing at least one display-only data element, one data entry element, one hidden data element, and one update-only data element for each of these data types: integer, floating point, alphanumeric, money, and date/time. ____ ____

2. Define a display area on the screen that remains hidden until needed, as opposed to an area that is visible on screen at all times. ____ ____

3. Define an array data entry area (i.e., multiple columns or rows with repeating data elements, such as may be found on a multiple-line-item purchase order or parts list). ____ ____

Total ____ ____

Average (Total/3) ____ ____ (3K)

3L. Ability to print screen layout document

Document the screen format by printing a copy of the screen commands or screen layout on a printer. ____ ____ (3L)

4.4 Report Writing Features

Generally, a 4GL provides a default report format for simple reports so that the user can issue a query and have the results printed as a report without having to specify elements of the report such as paging, date, page break, column position, etc. These attributes are tested first. In addition, the 4GL provides capabilities for specifying detailed report elements, if the user desires. Tasks are included for testing a number of these capabilities. Only those tasks that are required for the application need to be executed.

	Functionality	Performance/Ease of use
RATING SCALE	**0-none**	**0-not done/impossible**
	1-some	**1-poorly/hindrance**
	2-most	**2-adequately/helpful**
	3-all	**3-easily/very helpful**

4A. Default report formats provided for paging, dating, and headings

Using as many default instructions as possible, generate a report including at least two data elements. Specify only the data elements to be printed. Let the 4GL determine element placement, paging, and headings. ____ ____ (4A)

4B. Default report formats provided for summary functions

1. Generate a default report including a character element and at least one numeric data element and compute subtotals and totals on this numeric data element. ____ ____

2. Generate a default report that counts the number of records per group (subsort) and per report. ____ ____

3. Generate a default report that computes the average value of a numeric column by group and by report. ____ ____

Total ____ ____

Average (Total/3) ____ ____ (4B)

4C. Default report selections provided for ascending sorts
Generate a default report including at least one character data element and specify an alphabetic or ascending sort on this data element. ____ ____ (4C)

4D. Default report selections provided for descending sorts
Generate a default report including at least one numeric data element and specify a descending sort on this data element. ____ ____ (4D)

4E. Complex reporting options for row totals and subtotals
Create a report with multiple columns of numeric data elements. Generate and write row totals and subtotals. ____ ____ (4E)

4F. Reporting options for percent formats
Compute a percentage of two columns by using a built-in percent function and total the results. ____ ____ (4F)

4G. Reporting options for zero suppression
Display a number that is less than the maximum number of digits that can be displayed and suppress the leading zeros. ____ ____ (4G)

4H. Reporting options for floating dollar signs
Display a number in a numeric data element with currency format. Multiply the number by 10 and by 1000. Print the results. ____ ____ (4H)

4I. Comma option for numeric fields
Create a report with at least one numeric field. Display the field with commas separating every three digits. Display the field without commas. ____ ____ (4I)

4J. Reporting options for user specified spacing
Generate a report that is single spaced with a blank line after each subtotal and containing two blank lines after each total. ____ ____ (4J)

4K. Reporting options for user specified page breaks

1. Generate a report with a page break after each total. ____ ____

2. Generate a report with a page break after each subtotal. ____ ____

3. Generate a report with a page break after 40 lines. ____ ____

4. Generate a report with a page break after the value of a sort element has changed. ____ ____

Total ____ ____

Average (Total/4) ____ ____ (4K)

4L. Reporting options for table lookups

Create a report with a field that is referenced or cross-referenced from another file. (An example would be to list a column with the state name when the main file only has state codes.) ____ ____ (4L)

4M. Ability to generate report headings

1. Create a report with a header that appears only at the beginning of the report. ____ ____

2. Create a report with a header that is displayed at the top of each page of the report. ____ ____

3. Create a report with a header that is printed each time the grouping value changes. (i.e., for a number of records that are grouped by a common field value, a header that is displayed before each group). ____ ____

Total ____ ____

Average (Total/3) ____ ____ (4M)

4N. Ability to generate report footers

1. Create a report with a footer that is displayed only at the end of the report. ____ ____

2. Create a report with a footer that is printed at the bottom of each page of the report. ____ ____

3. Create a report with a footer that is printed after each group (i.e., for a number of records grouped by a common field value, a footer that is displayed after the last record in each group). ____ ____

Total ____ ____

Average (Total/3) ____ ____ (4N)

4O. Reporting options for font selection

Create a report with multiple fonts (e.g., large font for headings, smaller font for detail lines). ____ ____ (4O)

4P. Reporting options for odd size printouts

Create a report which exceeds the normal page size (e.g., 220 characters across the page and 90 lines per page). ____ ____ (4P)

4Q. Reporting options for forms printing
Create a form that incorporates line drawings with data from a database. Print the form with the data. ____ ____ (4Q)

4R. Reporting options for suppressed fields

1. Create a report that is sorted on one field. Sort the report by that field and suppress printing the field when repeated values occur. ____ ____

2. Using the same report, alter the setting so the sort field is printed on every occurrence. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (4R)

4S. Reporting options for calculated fields
Create a report with a numeric field value. Define a field in the report that is the result of some calculation involving the first field. ____ ____ (4S)

4T. Specific field placement
Define a report with specified field placement (i.e., the user specifies x-y coordinate or column-row values for field placement rather than using default values). ____ ____ (4T)

4U. Output device independence
Using any of the reports, print the report to the printer. Display the same report, without having to respecify the contents, on the screen. ____ ____ (4U)

4V. Output device adjustments
Using any report of at least 100 lines, print the report to the printer and display it on the screen. Did the 4GL adjust the page breaks, wrap arounds, headings, etc? ____ ____ (4V)

4.5 Language Features

The goal in this section is to test the language capabilities of the 4GL that are appropriate for your requirements. Keep in mind that the requirements may change over time. For example, a friendly novice level interface is nice in the beginning until users know the 4GL. After the 4GL has been used for a while, the novice level interface may become burdensome and cumbersome. It becomes aggravating to have to go through ten menus to select choices when you need only to enter three commands. Thus, you need to analyze the requirements to insure that the 4GL has the flexibility you will need not only from an initial perspective but also from a long range perspective.

This section mainly tests the query language or command language functions of a 4GL. Some 4GLs have a separate procedural language that is for developing programs, or have added syntax/commands for programming logic. We have tried to separate the two where possible. Technically the functions may be similar but the ease of use may be quite different. For example, some 4GLs have a condition statement that can be added, often by menu selection, to a query or report, e.g., IF code = 123, WHERE salary <50000, FOR name = "Smith", etc., which is different

from programming IF..THEN..ELSE logic to specify the execution of different sections of a program. The ease of use rating might be very different. The programming language functions are tested in the application development section.

	Functionality	Performance/Ease of use
RATING	**0-none**	**0-not done/impossible**
SCALE	**1-some**	**1-poorly/hindrance**
	2-most	**2-adequately/helpful**
	3-all	**3-easily/very helpful**

5A. SQL implementation

1. Complete tasks 5B through 5F using SQL commands. (See [FIPS127] "Database Language SQL.") Rate this task for overall implementation and integration with the rest of the 4GL. If you want to test other query capabilities of the 4GL (e.g., non-SQL interface) repeat these tasks and record a second set of numbers. ____ ____ (5A)

5B. Simple queries on one file

1. Query a file to find all records containing a specific non-null string. ____ ____

2. Query a file to find all records not containing a specific non-null string (the complement of the previous test). ____ ____

3. Query a file to find all records within a specified range. ____ ____

Total ____ ____

Average (Total/3) ____ ____ (5B)

5C. AND OR logic - one file

1. Query a file to find all records containing one data element value AND another element value. ____ ____

2. Query a file to find all records containing one data element value OR another element value. ____ ____

Total ____ ____ (5C)

Average (Total/2) ____ ____ (5C)

5D. Relational join

Relationally join two files without creating a third physically merged file. ____ ____ (5D)

5E. AND OR logic - two files

1. Query the joined files to find all records containing one data element value from one file AND one data element value from the other file (i.e., A AND B). ____ ____

2. Using the same joined files, query the files to find all records containing one data element value from one file OR another data element value from the other file (i.e., A OR B). ____ ____

Total ____ ____

Average (Total/2) ____ ____ (5E)

5F. Complex Boolean logic

1. Query the result of relationally joining three files to find all records containing one data element value from the first file AND one data element value from the second file OR one data element value from the third file (i.e., A AND (B OR C)). ____ ____

2. Query the same joined files for one data element value from one file OR one data element value from the second file AND one data element value from the third file (i.e., A OR (B AND C)). ____ ____

3. Query the joined files to find all records containing one data element value from one file AND one data element value from the second file AND NOT one data element value from the third file. Print or display the results showing data elements from all three files (i.e., A AND (B AND NOT C)). ____ ____

4. Query the same joined files for one data element value from one file OR one data element value from the second file AND NOT one data element value from the third file. Print or display the results showing multiple elements from all three files (i.e., A OR (B AND NOT C)). ____ ____

Total ____ ____

Average (Total/4) ____ ____ (5F)

5G. Traps command or statement errors

Enter a command or statement that contains a known identifiable error such as a keyword misspelling, illegal use of an operator, etc., to see how the 4GL traps and informs the user of the error. ____ ____ (5G)

5H. Facilitates correction of errors

Correct the error from the previous task and resubmit the command to see how the 4GL facilitates the correction of errors. ____ ____ (5H)

5I. Compiles code

Compile a number of commands or statements. (This may not be possible with some 4GLs that run only in interpretive mode. If so, rate functionality and ease as 0). ____ ____ (5I)

5J. Interprets code

Interpret a statement. (This may not be possible with some 4GLs that run only in compiled mode. If so, rate functionality and ease as 0.) ____ ____ (5J)

5K. Condition logic

1. Execute a simple condition statement with your query or report request. ____ ____

2. Execute a simple condition statement with alternative true and false outcomes. ____ ____

3. Execute a compound condition statement. ____ ____

4. Execute a compound condition statement that produces the complement result of the previous task. ____ ____

Total ____ ____

Average (Total/4) ____ ____ (5K)

5L. Looping logic

Execute an iterative loop structure such as repeating until the number of records returned equals 50. Do this as part of a query or report, i.e., do not program. ____ ____ (5L)

5M. Intrinsic date/time functions

1. Display the values of intrinsic date/time functions such as DATE, DAY, MONTH, DAY OF WEEK, DATE PLUS-OR-MINUS A TIME PERIOD, WEEK, QUARTER, YEAR, TODAY'S DATE, TIME. ____ ____

2. Store the results of intrinsic date/time functions in a database. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (5M)

5N. Intrinsic numeric functions

1. Display the values of intrinsic descriptive data functions such as MINIMUM, MAXIMUM, AVERAGE, COUNT, TOTAL, SUBTOTAL. ____ ____

2. Store the results of intrinsic descriptive data functions in a database. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (5N)

5O. Statistical functions

1. Perform the statistical calculations using intrinsic statistical routines if available. Check results for accuracy. Repeat task for each function required. (Add other functions as required by your application.)

Standard deviation ____ ____

Variance ____ ____

Regressional analysis ____ ____

Correlation ____ ____

Discriminant analysis ____ ____

Factor analysis ____ ____

Descriptive statistics ____ ____

2. Store the statistical results generated in #1 to a database. Repeat for each function required. (Add other functions as required.)

Standard deviation ____ ____

Variance ____ ____

Regressional analysis ____ ____

Correlation ____ ____

Discriminant analysis ____ ____

Factor analysis ____ ____

Descriptive statistics ____ ____

Total ____ ____

Average (Total/#) ____ ____ (5O)

5P. Ability to process missing data

1. Create or use a file with missing values in a numeric field. Compute the average of this numeric field including the records with missing values. ____ ____

2. Using the same file, average the numeric field excluding the records with missing values. ____ ____

3. Compare the two results. If the 4GL has correctly processed missing values the average excluding the missing value records will be higher than the average including the missing value records. If it correctly processed the missing values score a three for functionality. If it did not, score zero. ____ ____

Total ____ ____

Average (Total/3) ____ ____ (5P)

4.6 Graphics Features

The goal of this section is to evaluate the graphics features that are incorporated in the 4GL. Some 4GLs actually link to separate graphics software while others have integrated all of the graphics functions with the database and language functionality of the 4GL. You need to evaluate the ease of use of various features, how much effort, rekeying, etc., it takes to generate certain kinds of graphs. You also need to evaluate the quality of the graphs generated on the screen and in a printed version, discounting the quality of the printer itself.

	Functionality	Performance/Ease of use
RATING SCALE	0-none	0-not done/impossible
	1-some	1-poorly/hindrance
	2-most	2-adequately/helpful
	3-all	3-easily/very helpful

6A. Bar graph
Create a bar graph from data in the database. ____ ____ (6A)

6B. Pie chart
Create a pie chart from data in the database. ____ ____ (6B)

6C. Exploded pie chart
Create an exploded pie chart from data in the database. ____ ____ (6C)

6D. Scatter diagram
Create a scatter diagram from data in the database. ____ ____ (6D)

6E. Line graph
Create a line graph from data in the database. ____ ____ (6E)

6F. Connected point plot
Create a connected point plot from data in the database. ____ ____ (6F)

6G. Trend line
Create a scatter diagram that incorporates a trend line. ____ ____ (6G)

6H. Curve fitting
Create a graph that requires curve fitting or smoothing. ____ ____ (6H)

6I. Automatic scaling
Create one graph with data points and a second graph with the data points multiplied by 1000 to demonstrate automatic scaling. ____ ____ (6I)

6J. Logarithmic scaling
Create a graph whose data points would require logarithmic scaling. ____ ____ (6J)

6K. Labels and titles
Generate field labels and titles for any graph from the data dictionary (do not reenter labels and titles). ____ ____ (6K)

6L. Multiple fonts
Create a graph which utilizes multiple fonts (one font for titles, another for labels, etc.) ____ ____ (6L)

6M. Calculated values
Create a line graph with one line representing computed values (such as a ratio of two fields in the database). ____ ____ (6M)

6N. Grid option
Add a grid pattern to any graph. ____ ____ (6N)

6O. Color option
Add graph colors for either print or display. ____ ____ (6O)

6P. Hatch pattern options
Change the hatch patterns for any bar graph. ____ ____ (6P)

6Q. Ability to process missing values
Create a line graph representing a statistic that includes processing missing values. ____ ____ (6Q)

4.7 Security Features

These tasks test some of the basic security features of most 4GLs. Most of these features are implemented very differently from product to product. Some 4GLs have separate workspaces identified for each user so identification and password protection are done at this level. Others require identifiers and passwords for each file or procedure. Some 4GLs have various levels of security allowing quite complex security while others provide only read/write protection for files.

These tasks were not designed to represent ALL of the security tasks that could be tested. They were designed to test the more common kinds of security that is generally provided by 4GLs. You may want to add additional tasks if additional levels of security are required for your application. For example, to provide a minimum test for encrypted files, you may want to run an analysis of the character distribution of the files before and after encryption. You may also want to tailor these tasks to your particular application. For example, the section on command security could be altered to test protection from use of specific commands or programs that are of interest for your application.

One other important part of security has not been included in these tasks, backup and recovery. Since backup and recovery are often dependent on other facilities of the hardware and software environment it is difficult to generalize tasks that could run on microprocessor versions of the 4GL and be an indication of mainframe or minicomputer versions of the same 4GL. You could turn off the power in the middle of running a query to analyze what happens to the data but this would not necessarily indicate what would happen with the same 4GL in a different operating environment. Some 4GLs running on mainframes or minis will mark data that has been involved in a crash, provide rollback facilities, and issue warnings that there has been a crash. There are few that offer this protection in the microcomputer versions. Networked versions of the 4GL also offer different facilities than stand-alone micro versions.

Since this area cannot be tested as part of the functional benchmarks, we recommend that you talk to the vendor and other users of the 4GL under consideration, to gather information about backup and recovery on the hardware and software environment you require. Actual case studies from other users of what happened when the power went off or when the disk crashed would be invaluable.

	Functionality	Performance/Ease of use
RATING	**0-none**	**0-not done/impossible**
SCALE	**1-some**	**1-poorly/hindrance**
	2-most	**2-adequately/helpful**
	3-all	**3-easily/very helpful**

7A. Specify user access

Grant unlimited file rights to a specified user. Access the file as the user. Attempt to access the file as a different user to ensure that access is denied. ____ ____ (7A)

7B. Specify file security

1. Limit a user's file rights to read only privileges. Display the file. Attempt to add a record to see if the request is denied. Attempt to delete a record; to edit a record. ____ ____

2. Establish write privileges for a user. View the file. Add a record. Edit data in a record. Delete a record. ____ ____

3. Deny access to a file for read or write privileges for a user. Access the system as this user. Attempt to read or query the file to verify that this is not possible. Obtain a listing or access a screen with the database files listed to ensure that the file is not listed. ____ ____

Total ____ ____

Average (Total/3) ____ ____ (7B)

7C. Limit command or procedure usage

1. Deny a user's rights to update a file. Access the system as the user. Attempt to change or modify a record to verify that this is not possible. ____ ____

2. Grant a user the right to retrieve no more than 10 records from any query. Attempt to display more than 10 records to determine that this is not permissible. ____ ____

3. Create a command procedure, subroutine or program to make a listing of the files in the database. Set the security to prevent a user from running this procedure. Access the system as this user and attempt to run the procedure to ensure that this is not possible. ____ ____

4. Block a user from using the program editor to change any program (restrict the use of the EDIT command). Access the system as this user and attempt to edit a program file. ____ ____

Total ____ ____

Average (Total/4) ____ ____ (7C)

7D. Specify record access

1. Place limitations on the accessibility of certain records in a file. Display the file to make sure restricted records are not shown. ____ ____

2. Join the file with limited record access to a file that has unlimited access. Display the contents of the joined files to see if the records remain hidden. (A score of three should be given if the record remains hidden, zero if the record is now accessible.) ____ ____

Total ____ ____

Average (Total/2) ____ ____ (7D)

7E. Specify field access

1. Limit the access capabilities of one field in a file. Display the contents of the file to make certain the field does not show. If possible, list the structure or dictionary of the file to see if the field is hidden. ____ ____

2. Join the file with a limited field to a file that has unlimited access. Display the contents to see if the field is still hidden. (A score of three should be given if the field remains hidden, zero if the field is now accessible). ____ ____

3. Grant a user the right to update only certain fields in a file (i.e., some fields cannot be updated). ____ ____

Total ____ ____

Average (Total/3) ____ ____ (7E)

7F. Encrypt/decrypt files

1. Encrypt a data file. Attempt to view the file (type or list from the operating system) without decrypting to see if the file appears readable. ____ ____

2. Decrypt the data file. View the file to make certain the data is readable. Compare with original data to ensure the file contents were not altered. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (7F)

7G. Encrypt/decrypt data dictionary

1. Encrypt the data dictionary. Attempt to view the data dictionary to see if it appears readable. ____ ____

2. Decrypt the data dictionary. Compare with the original data dictionary to make certain it is unchanged. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (7G)

7H. Default logon procedure

Create a procedure or program that is automatically executed whenever the 4GL is started. Have the logon procedure or profile tailor the system to the user's requirements. ____ ____ (7H)

7I. User profile

Define a user profile to limit access to the system. (i.e., accessing the 4GL requires the user to enter a user ID and/or password before access to the system is granted. Only acceptable ID's/passwords are granted user privileges). Note: Some 4GLs do this automatically, some have to be programmed. ____ ____ (7I)

7J. Define system administrator

Establish a user ID and/or password for a user or users that grants this user the right to establish IDs, passwords, file access, etc., for other users. Evaluate the protection for this user ID and password. (Are the ID and password encrypted? Are they missing from the default listing of users, etc?) ____ ____ (7J)

7K. Feature restrictions

1. Restrict a user, based on user ID and/or password, to only one part of the 4GL (e.g., the report writer). Log in to the 4GL as the user and attempt to access other parts of the 4GL (e.g., screen generator) to determine if access is denied. ____ ____

2. Change a user's ID/password without having to redefine the restrictions imposed on the user. Log in to the 4GL with the new ID/password and make certain the same restrictions are still valid. ____ ____

3. Delete the user's ID/password. Attempt to log in to the 4GL with the password to make sure access is now denied. ____ ____

Total ____ ____

Average (Total/3) ____ ____ (7K)

7L. Logon audit file

Create a file that records the name of the user, the time and date the user logged on, and the time the user logged off. Display the user logon file and verify that the contents are correct. ____ ____ (7L)

4.8 Implementation Features

A number of the tasks grouped under implementation features, such as rating the documentation, on-line help, diagnostics and messages, are not really rating one specific task but rating the usage of these features in general. Others require interfacing to other languages or other software. Whatever interfaces are required by your application should be used for these tasks.

	Functionality	Performance/Ease of use
RATING	**0-none**	**0-not done/impossible**
SCALE	**1-some**	**1-poorly/hindrance**
	2-most	**2-adequately/helpful**
	3-all	**3-easily/very helpful**

8A. Interface to other software packages

1. Interface to an external software package (a graphics package, a statistical analysis package, etc.) without having to exit the 4GL with a "quit" or "end" type of command. ____ ____

2. Move data from the 4GL to the other package and display the resulting data. (Did the data dictionary information transfer or did the data have to be redefined? Were the data types correct, i.e. integers stayed integers, numeric were not changed to character, etc.) ____ ____

Total ____ ____

Average (Total/2) ____ ____ (8A)

8B. Interface to other languages

Execute a procedure or program written in a different language (Fortran, C, another 4GL) without having to exit the 4GL with a "quit" or "end" type of command. (The execution command can be in a command mode if available or imbedded in a 4GL program.) ____ ____ (8B)

8C. Interface to operating system

Access the operating system (i.e., such as listing a file directory) without having to exit the 4GL with a "quit" or "end" type of command. ____ ____ (8C)

8D. POSIX compatibility

Execute the 4GL in a POSIX conforming environment. (See [FIPS151], "Portable Operating System Interface for Computer Environments.") ____ ____ (8D)

8E. On-line HELP

Access on-line help from the 4GL and display it on the screen. Also remember if you used the on-line help for other benchmarking tasks and if it was helpful. (If on-line help is not available, rate this zero.) ____ ____ (8E)

8F. Documentation

Rate the helpfulness of the documentation. Was it up to date with the current version? Was it complete? Did the 4GL do what the documentation said it would do? Was it easy to understand? ____ ____ (8F)

8G. Interface to external editor/word processor

1. Generate commands, programs, or file descriptions, etc. in a word processor or different program editor. Execute the code to ensure the 4GL accepts it. ____ ____

2. Utilize an external editor or word processor without having to exit the 4GL with a "quit" or "end" type of command. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (8G)

8H. Error messages and warnings

1. Try to issue a query or report request with an illegal field name. Was an error message displayed that was meaningful and helpful? ____ ____

2. Based on experience from other tasks, are warning messages displayed informing the user that a query or other task might take a long time, that a certain action might delete a file, etc.? ____ ____

Total ____ ____

Average (Total/2) ____ ____ (8H)

4.9 Application Development

There are certain parts of the 4GLs that are difficult to assess in a piecemeal fashion. A number of these are part of the process of actually building an application. The pieces of an application, such as screens, reports, etc., can be tested as we have listed in appropriate sections of this report. What is more difficult to specify in individual tasks is the functionality of the 4GL to facilitate putting these pieces together to build an application. This functionality incorporates the robustness of the programming or procedural language with the ability to integrate the language with other segments that may have been generated by the nonprocedural capabilities of the 4GL.

Some 4GL products have been joined with CASE products or other software design tools. This section is not going to suggest rating these capabilities because they are outside of the scope of most 4GLs. This section also does not attempt to retest those capabilities that have been listed in other sections of this report. Instead it focuses on areas that cannot be tested except by building an application. For example, some 4GLs have conditional logic that can be used in report or query requests (IF a value = x, WHERE a code is less than some number, IF the report does not exceed

a record limit, etc.). This logic and syntax is sometimes quite different from logic which indicates branching to different parts of a program (IF something occurs then execute a subroutine, or process a different section of the program).

We suggest that you design and build a very small application that would be essentially a small prototype of your application, including representative files, reports and queries. The application that we used, described below, was developed in an average of four hours using several different 4GL products. The application needs to be developed before the tasks can be rated.

EXAMPLE APPLICATION

Our application was based on two very small (only 10 records each) files. One file had an identifier (employee identification number), employee name, and salary. The other file had cost centers, identifiers, and number of hours. The basic idea was that most applications would have some file modifications (changes to records, adding records, or deleting records) and some queries or reports. Our application required posting of hours to modify the database, and reporting the records by joining the files and multiplying the salary times the number of hours. We reported the records by employee and by cost center. A general description is given in figure 2.

Figure 3 gives the description and contents of the two files that were used for the application. These files were kept very simple since these tasks were not designed to test the file handling capabilities of the 4GL, but were designed to test the procedural language capabilities. Figure 4 is a sample of the kinds of screens that we developed and figure 5 is a sample of one of the reports. We have generalized these figures to represent the results of many 4GLs. They are not actual printouts of any specific 4GL product.

Even a small application tests some of the procedural language capabilities of the 4GL. The program must be able to provide the three basic constructs of sequence, iteration, and selection. The program can incorporate screens, menus, or reports that come from the nonprocedural part of the 4GL but the screens, menus, and reports must be integrated with programming logic to create a stand-alone application. We also included calculations and dollar figures with floating dollar signs because these are handled differently by many 4GLs and are also typical of most business applications.

After a small application has been developed, you should be able to give ratings to the following tasks. We strongly suggest that you use an application that is typical of your large application or a small prototype of your application. The requirements should be kept to a minimum, however, to prevent these tasks from being overburdensome. If you have specialized programming requirements, then we suggest that you add tasks to this area.

	Functionality	Performance/Ease of use
RATING	0-none	0-not done/impossible
SCALE	1-some	1-poorly/hindrance
	2-most	2-adequately/helpful
	3-all	3-easily/very helpful

9A. Program editor

Rate the functionality and ease of use of the 4GL program editor. ____ ____ (9A)

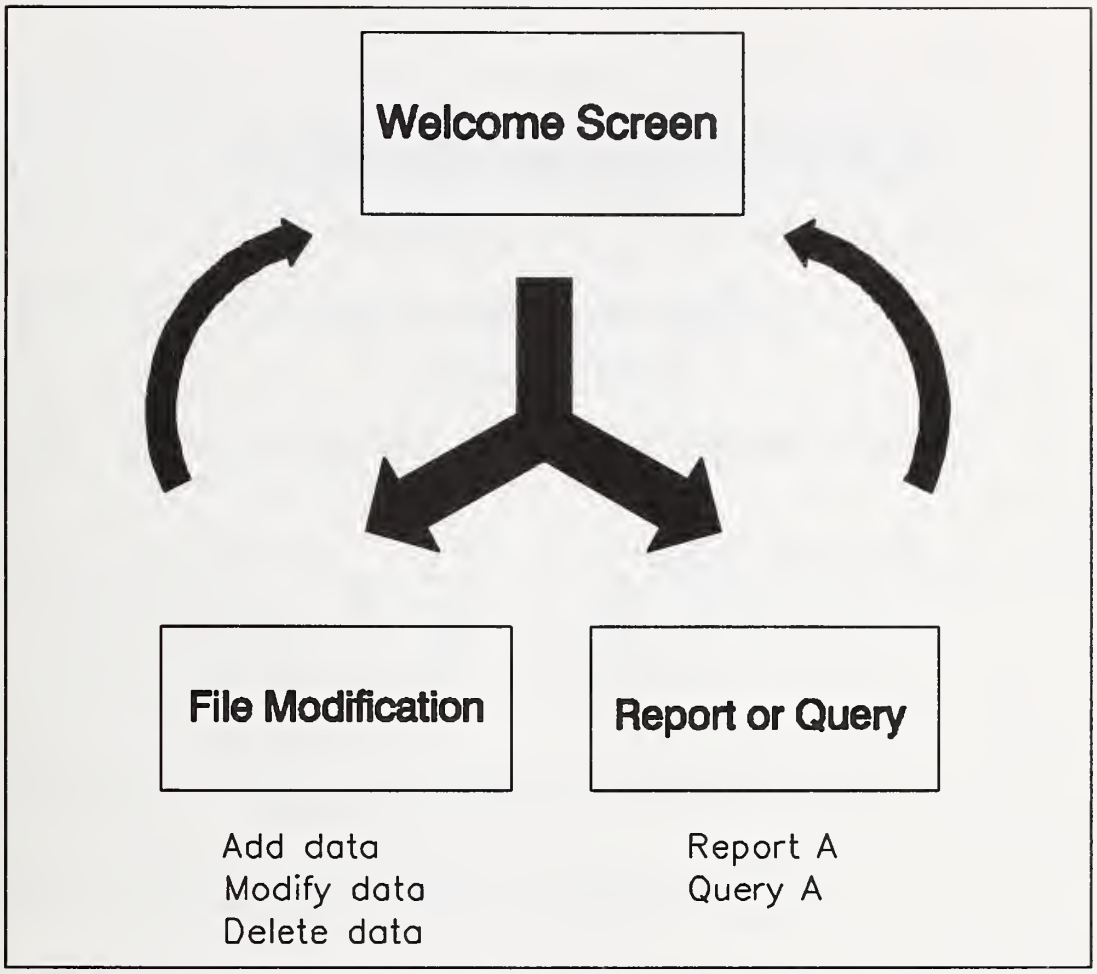


Figure 2. Test Application

TEST APPLICATION FILES

DESCRIPTION
Filename:EMPLOYEE
Field 1: ID - 4 characters
Field 2: LASTNAME - 10 characters
Field 3: FIRSTNAME - 10 characters
Field 4: SALARY - 5 numbers with
2 decimal places

CONTENTS

ID	LASTNAME	FIRSTNAME	SALARY
107	Madison	Michele	07.82
210	Eisenhower	Murph	07.82
312	Jones	James	09.68
413	Truman	Elizabeth	09.68
514	Wilson	John	11.85
610	Swivit	Rodney	13.04
715	Harrison	Suzanne	14.33
807	Roosevelt	John	17.17
907	Garfield	Elizabeth	20.42
1015	Harrison	James	24.13

DESCRIPTION
Filename:COSTCEN
Field 1: ID - 4 characters
Field 2: CEN - 4 characters
Field 3: HRS - 3 numbers with
0 decimal places

CONTENTS

ID	CEN	HRS
107	0101	0
210	0202	0
312	0101	0
413	0202	0
514	0301	0
610	0301	0
715	0101	0
807	0401	0
907	0202	0
1015	0101	0

Figure 3. Application Files

WELCOME TO THE 4GL TEST SYSTEM!

YOU MAY: 1) Post hours for employees

2) Report by employee or cost center

3) Exit the system

MAKE YOUR SELECTION (enter 1, 2, or 3):

Figure 4. Application Screen

TEST APPLICATION REPORT SAMPLE

REPORT BY COST CENTER

CEN	ID	LASTNAME	FIRSTNAME	HRS	DOLLARS
101	107	Madison	Michele	26	$203.32
210	Eisenhower	Murph	20	$156.40	
715	Harrison	Susanne	5	$71.65	
1015	Harrison	James	22	$530.86	
TOTAL FOR COST CENTER			73	$962.23	
202	413	Truman	Elizabeth	35	$338.80
210	Eisenhower	Murph	42	$328.44	
907	Garfield	Elizabeth	24	$490.08	
TOTAL FOR COST CENTER			101	$1157.32	
301	514	Wilson	John	52	$616.20
610	Swivit	Rodney	8	$104.32	
TOTAL FOR COST CENTER			60	$720.52	
401	715	Harrison	Susanne	12	$171.96
TOTAL FOR COST CENTER			12	$171.96	
TOTAL				246	$3012.03

Figure 5. Application Report

9B. Error messages
Rate how helpful the error messages were. When you ran the application, did you get messages that merely said there is an error #xxx, or did it explain what the error was and perhaps contain syntax suggestions, line numbers of the errors, etc? ____ ____ (9B)

9C. Branching logic
Were you able to move to different sections of the program to reflect choice selection? ____ ____ (9C)

9D. Looping logic
Could you execute a process multiple times using iteration, (e.g., multiple reports, multiple records, etc.)? ____ ____ (9D)

9E. Language integration
Was the programming language consistent with the language you used for commands or similar to the language of menus or screens provided in the nonprocedural parts of the 4GL? If the syntax and functionality are consistent across the 4GL rate the functionality three. ____ ____ (9E)

9F. Ability to call subroutines
1. Enter a block of statements and store them as a procedure or program (PROCEDURE A). Execute PROCEDURE A. ____ ____

2. Enter another stored procedure (PROCEDURE B), one statement of which executes PROCEDURE A. Execute PROCEDURE B. ____ ____

Total ____ ____

Average (Total/2) ____ ____ (9F)

9G. Ability to imbed queries in procedures
Create a report procedure, REPORT A, containing a conditional statement to compute a SUM and statements to query a data file for a subset of the file's data. ____ ____ (9G)

9H. Ability to share temporary data
Execute a procedure that extracts a temporary subset of data from a file and makes this data available to another stored procedure within the 4GL. ____ ____ (9H)

9I. Ability to create global variables
Define global memory variables (temporary variables which are accessible to more than one program). ____ ____ (9I)

9J. Ability to create local variables
Define local memory variables (temporary variables which are accessible only to one program). ____ ____ (9J)

9K. Ability to define calculations using field values
Define a memory variable that is a result of a calculation between two fields. ____ ____ (9K)

5. 4GL BENCHMARKING SUMMARY

The benchmarking summary sheet can be used for the final step in evaluating specific products. The requirements weights determined how important a feature was to the application. The scores on the functional benchmark tasks indicated how well the 4GL exhibited the feature and how easy it was to perform the task or tasks for the feature. Taken together the results can assist a decision maker in the final selection of a 4GL product.

The weights from the requirements checklist and the ratings or average ratings of the benchmark tasks should be transferred to this summary sheet. An overall rating can be obtained by multiplying the weights times the sum of the task points and summing these points for each product evaluated. An example of this summary process is given below using part of data management as an illustration.

STEP ONE - TRANSFER REQUIREMENT WEIGHTS

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
A. Basic data dictionary	5			
B. Synonym or aliases	1			
C. Headings, titles, or labels	4			
D. Comments or remarks	3			
E. Date formats	4			
F. Time formats	0			

STEP TWO - TRANSFER BENCHMARK TASK SCORES

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
A. Basic data dictionary	5	3	2.5	
B. Synonym or aliases	1	3	1	
C. Headings, titles, or labels	4	0	0	
D. Comments or remarks	3	3	3	
E. Date formats	4	2	3	
F. Time formats	0	0	0	

STEP THREE - MULTIPLY WEIGHT TIMES SUM OF FUNC AND EASE - SUM ALL

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
A. Basic data dictionary	5	3	2.5	27.5
B. Synonym or aliases	1	3	1	4
C. Headings, titles, or labels	4	0	0	0
D. Comments or remarks	3	3	3	18
E. Date formats	4	2	3	20
F. Time formats	0	0	0	0
				TOTAL 69.5

4GL SUMMARY SHEET

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
1. OPERATIONAL FEATURES				
A. Installation management	_____	___	___	_____
B. Ease of installing on multiple platforms	_____	___	___	_____
2. DATA MANAGEMENT				
A. Basic data dictionary	_____	___	___	_____
B. Synonym or aliases	_____	___	___	_____
C. Headings, titles, or labels	_____	___	___	_____
D. Comments or remarks	_____	___	___	_____
E. Date formats	_____	___	___	_____
F. Time formats	_____	___	___	_____
G. Money formats	_____	___	___	_____
H. Variable-length fields	_____	___	___	_____
I. Capable of handling _____ # chars per field	_____	___	___	_____

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
J. Capable of handling _____# fields per record	_____	____	____	_____
K. Keyed or indexed files	_____	____	____	_____
L. Data structure management	_____	____	____	_____
M. Ability to read, query, or report from ASCII files	_____	____	____	_____
N. Ability to import ASCII files	_____	____	____	_____
O. Ability to export ASCII files	_____	____	____	_____
P. Ability to read, query, or report from _____ formatted files	_____	____	____	_____
Q. Ability to import _____ formatted files	_____	____	____	_____
R. Ability to export _____ formatted files	_____	____	____	_____
S. Ability to modify files while loading	_____	____	____	_____
T. File check while loading	_____	____	____	_____
U. File modification	_____	____	____	_____
3. USER INTERFACE				
A. Capability to create simple menus	_____	____	____	_____
B. Capability to create complex menus	_____	____	____	_____
C. Default data entry screens provided	_____	____	____	_____
D. Capability to create straight text screens	_____	____	____	_____
E. Capability to create display screens	_____	____	____	_____
F. Data checking on data entry	_____	____	____	_____
G. Reverse video control	_____	____	____	_____

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
H. Blinking data entry fields	_____	____	____	_____
I. Calculated fields display	_____	____	____	_____
J. Display of HELP information on screen	_____	____	____	_____
K. Capability to generate more complex screens	_____	____	____	_____
L. Ability to print screen layout document	_____	____	____	_____
4. REPORT WRITING FEATURES				
A. Default report formats provided for paging, dating, and headings	_____	____	____	_____
B. Default report formats provided for summary functions	_____	____	____	_____
C. Default report selections provided for ascending sorts	_____	____	____	_____
D. Default report selections provided for descending sorts	_____	____	____	_____
Complex reporting options for:				
E. Row totals and subtotals	_____	____	____	_____
F. Percent formats	_____	____	____	_____
G. Zero suppression	_____	____	____	_____
H. Floating dollar signs	_____	____	____	_____
I. Comma option for numeric fields	_____	____	____	_____
J. User specified spacing	_____	____	____	_____
K. User specified page breaks	_____	____	____	_____
L. Table lookups	_____	____	____	_____
M. Headings	_____	____	____	_____
N. Footers	_____	____	____	_____

		Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
O.	Font selection	_____	___	___	_____
P.	Odd size printouts	_____	___	___	_____
Q.	Forms printing	_____	___	___	_____
R.	Suppressed fields	_____	___	___	_____
S.	Calculated fields	_____	___	___	_____
T.	Specific field placement	_____	___	___	_____
U.	Output device independence (i.e., user could select screen, printer, or file)	_____	___	___	_____
V.	Output device adjustments (i.e., user selects screen, printer, or file and 4GL adjusts for page size, screen size, etc.)	_____	___	___	_____
5.	LANGUAGE FEATURES				
A.	SQL implementation	_____	___	___	_____
B.	Simple queries on one file	_____	___	___	_____
C.	AND OR logic - one file	_____	___	___	_____
D.	Relational join	_____	___	___	_____
E.	AND OR logic - two files	_____	___	___	_____
F.	Complex Boolean logic	_____	___	___	_____
G.	Traps command or statement errors	_____	___	___	_____
H.	Facilitates correction of errors	_____	___	___	_____
I.	Compiles code	_____	___	___	_____
J.	Interprets code	_____	___	___	_____
K.	Condition logic	_____	___	___	_____
L.	Looping logic	_____	___	___	_____

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
M. Intrinsic date/time functions (DATE, DAY, MONTH, etc.)	_____	___	___	_____
N. Intrinsic numeric functions (MINIMUM, MAXIMUM, AVERAGE, etc.)	_____	___	___	_____
O. Statistical functions:				
Standard deviation	_____	___	___	_____
Variance	_____	___	___	_____
Regressional analysis	_____	___	___	_____
Correlation	_____	___	___	_____
Discriminant analysis	_____	___	___	_____
Factor analysis	_____	___	___	_____
Descriptive statistics	_____	___	___	_____
P. Ability to process missing data	_____	___	___	_____
6. GRAPHICS FEATURES				
A. Ability to create bar graphs	_____	___	___	_____
B. Ability to create pie charts	_____	___	___	_____
C. Ability to create exploded pie charts	_____	___	___	_____
D. Ability to create scatter diagrams	_____	___	___	_____
E. Ability to create line graphs	_____	___	___	_____
F. Ability to create connected point plots	_____	___	___	_____
G. Ability to incorporate trend lines in scatter diagrams	_____	___	___	_____
H. Ability to curve fit or smooth	_____	___	___	_____
I. Availability of automatic scaling	_____	___	___	_____
J. Availability of logarithmic scaling	_____	___	___	_____

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
K. Ability to generate field labels and titles from the data dictionary	_____	___	___	_____
L. Ability to utilize multiple fonts	_____	___	___	_____
M. Ability to calculate values to incorporate in the graph	_____	___	___	_____
N. Grid option	_____	___	___	_____
O. Color option	_____	___	___	_____
P. Hatch pattern options	_____	___	___	_____
Q. Ability to process missing values	_____	___	___	_____
7. SECURITY FEATURES				
A. Ability to specify user access	_____	___	___	_____
B. Ability to specify file security	_____	___	___	_____
C. Ability to limit command or procedure usage	_____	___	___	_____
D. Ability to specify record access	_____	___	___	_____
E. Ability to specify field access	_____	___	___	_____
F. Ability to encrypt/decrypt files	_____	___	___	_____
G. Ability to encrypt/decrypt data dictionary	_____	___	___	_____
H. Default logon procedure	_____	___	___	_____
I. User profile	_____	___	___	_____
J. Ability to define system administrator	_____	___	___	_____
K. Feature restrictions	_____	___	___	_____
L. Ability to create a user logon audit file	_____	___	___	_____

	Weight	FUNC	EASE	Wt. mult. by FUNC+EASE
8. IMPLEMENTATION FEATURES				
A. Interface to other software packages	_____	____	____	_____
B. Interface to other languages	_____	____	____	_____
C. Interface to operating system	_____	____	____	_____
D. POSIX compatibility	_____	____	____	_____
E. Online HELP	_____	____	____	_____
F. Documentation	_____	____	____	_____
G. Interface to external editor/ word processor	_____	____	____	_____
H. Error messages and warnings	_____	____	____	_____
9. APPLICATION DEVELOPMENT				
A. Program editor	_____	____	____	_____
B. Error messages	_____	____	____	_____
C. Branching logic	_____	____	____	_____
D. Looping logic	_____	____	____	_____
E. Language integration	_____	____	____	_____
F. Ability to call subroutines	_____	____	____	_____
G. Ability to imbed queries in procedures	_____	____	____	_____
H. Ability to share temporary data	_____	____	____	_____
I. Ability to create global variables	_____	____	____	_____
J. Ability to create local variables	_____	____	____	_____
K. Ability to define calculations using field values	_____	____	____	_____

6. ANALYSIS AND CONCLUSIONS

After the complete functional benchmarking process has been repeated for each 4GL product under consideration, the total numbers can be compared. A higher total number for one product compared to the total of another product indicates that one product more fully met the requirements of an application than the other.

The ratings of the individual tasks, especially for the highest weighted requirements, should also be analyzed. This is important when two products have total scores that are very close to each other. Analyzing individual functional areas of the 4GLs reveal the strengths and weaknesses of the products relative to the requirements. Because certain requirements have more than one task, analysis of the worksheets for each task may also be informative. An analysis of variance of individual area scores may tell you how much confidence you should have in your choice.

Completion of the tasks and the evaluation of the results may also reveal that the initial requirements weights should be adjusted. Sometimes requirements that seemed unimportant to an application are discovered to be more important than originally thought. This adjustment of requirements is an important part of the prototyping process. Completing the functional benchmarking tasks is similar to completing a small prototype. A more thorough understanding of the application's requirements is gained in the process.

The testing methodology and the set of functional benchmark tasks provided in this report should facilitate the evaluation of 4GL products in relation to organizational requirements. Completing the tasks requires some effort but the results will provide a more thorough understanding of the 4GL products under consideration. This should allow federal managers to make better informed decisions in the selection and procurement process for a 4GL.

REFERENCES

[FIPS42] **Guidelines for Benchmarking ADP Systems in the Competitive Procurement Environment**, Federal Information Processing Standards Publication 42-1 (FIPS PUB 42-1), National Bureau of Standards, Gaithersburg, MD, May 15, 1977.

[FIPS127] **Database Language SQL**, Federal Information Processing Standards Publication 127-1 (FIPS PUB 127-1), National Institute of Standards and Technology, Gaithersburg, MD, February 2, 1990.

[FIPS151] **POSIX: Portable Operating System Interface for Computer Environments**, Federal Information Processing Standards Publication 151-1 (FIPS PUB 151-1), National Institute of Standards and Technology, Gaithersburg, MD, March 28, 1990.

[FISH86] Fisher, Gary E., **A Functional Model for Fourth Generation Languages**, NBS Special Publication 500-138, National Bureau of Standards, Gaithersburg, MD, June 1986, 28p. 23 refs.

[FISH87] Fisher, Gary E., **Application Software Prototyping and Fourth Generation Languages**, NBS Special Publication 500-148, May 1987, 59p. 44 refs.

[GRAY86] Gray, Martha M., **Guide to the Selection and Use of Fourth Generation Languages**, NBS Special Publication 500-143, September 1986, 58p. 100 refs.

INDEX

NIST-114A
(REV. 3-89)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

BIBLIOGRAPHIC DATA SHEET

1. PUBLICATION OR REPORT NUMBER
NIST/SP-500/184

2. PERFORMING ORGANIZATION REPORT NUMBER

3. PUBLICATION DATE
March 1991

4. TITLE AND SUBTITLE

Functional Benchmarks for Fourth Generation Languages

5. AUTHOR(S)

Martha Mulford Gray and Gary E. Fisher

6. PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
GAITHERSBURG, MD 20899

7. CONTRACT/GRANT NUMBER

8. TYPE OF REPORT AND PERIOD COVERED
Final

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)

Software Engineering Group
Systems and Software Technology Division
National Computer Systems Laboratory NIST
Gaithersburg, Maryland 20899

10. SUPPLEMENTARY NOTES

[ ] DOCUMENT DESCRIBES A COMPUTER PROGRAM; SF-185, FIPS SOFTWARE SUMMARY, IS ATTACHED.

11. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

In recent years, fourth generation language (4GL) usage has expanded in data processing organizations, especially where end-users have assumed increased programming responsibilities. Since there are no 4GL standards, managers selecting a 4GL need some method of determining how well a particular 4GL will meet organizational, application, and user requirements. This report provides a methodology to assist in that determination. It contains "functional benchmarks" (as opposed to performance benchmarks) consisting of a testing methodology and descriptions of tests to evaluate the capabilities of a particular 4GL in relation to organizational requirements. Test results are evaluated at two levels: the 4GL's ability to perform a task, and the ease of performing it. The evaluations are combined with user-defined weighted requirements to produce an overall rating for each 4GL tested.

12. KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)

database management; end-user computing; 4GL, 4GL requirements, fourth generation language, functional benchmark, nonprocedural language; report writer; screen generation.

13. AVAILABILITY

[X] UNLIMITED

[ ] FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).

[X] ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.

[X] ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES
60

15. PRICE

ELECTRONIC FORM

*U.S.G.P.O:1991-281-557:40023

ANNOUNCEMENT OF NEW PUBLICATIONS ON COMPUTER SYSTEMS TECHNOLOGY

Superintendent of Documents
Government Printing Office
Washington, DC 20402

Dear Sir:

Please add my name to the announcement list of new publications to be issued in the series: National Institute of Standards and Technology Special Publication 500-.

Name _____

Company _____

Address _____

City _____ State _____ Zip Code _____

(Notification key N-503)

NIST Technical Publications

Periodical

Journal of Research of the National Institute of Standards and Technology—Reports NIST research and development in those disciplines of the physical and engineering sciences in which the Institute is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Institute's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Institute's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NIST, NIST annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NIST under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published quarterly for NIST by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW., Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Institute on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NIST under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NIST administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NIST research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order the **above** *NIST publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.*

Order the **following** *NIST publications—FIPS and NISTIRs—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NIST pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NIST Interagency Reports (NISTIR)—A special series of interim or final reports on work performed by NIST for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Institute of Standards and Technology
Gaithersburg, MD 20899